US009087301B2

(12) **United States Patent**
Alvarez-Icaza Rivera et al.

(10) **Patent No.: US 9,087,301 B2**
(45) **Date of Patent: Jul. 21, 2015**

(54) **HARDWARE ARCHITECTURE FOR SIMULATING A NEURAL NETWORK OF NEURONS**

(71) Applicant: **INTERNATIONAL BUSINESS MACHINES CORPORATION**, Armonk, NY (US)

(72) Inventors: **Rodrigo Alvarez-Icaza Rivera**, San Jose, CA (US); **John V. Arthur**, Mountain View, CA (US); **Andrew S. Cassidy**, San Jose, CA (US); **Pallab Datta**, San Jose, CA (US); **Paul A. Merolla**, Palo Alto, CA (US); **Dharmendra S. Modha**, San Jose, CA (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 165 days.

(21) Appl. No.: **13/725,452**

(22) Filed: **Dec. 21, 2012**

(65) **Prior Publication Data**

US 2014/0180988 A1 Jun. 26, 2014

(51) **Int. Cl.**

G06N 3/063	(2006.01)
G06N 3/04	(2006.01)
G06N 3/06	(2006.01)

(52) **U.S. Cl.**

CPC ***G06N 3/049*** (2013.01); ***G06N 3/063*** (2013.01); *G06N 3/06* (2013.01); *G06N 3/0635* (2013.01)

(58) **Field of Classification Search**

None

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,999,953	B2	2/2006	Ovhsinsky
7,870,086	B2	1/2011	Llinas et al.
7,908,235	B2	3/2011	Snook et al.
8,200,593	B2	6/2012	Guillen et al.
2014/0180985	A1	6/2014	Alvarez-Icaza Rivera et al.

FOREIGN PATENT DOCUMENTS

EP	0560595	A2	9/1993

OTHER PUBLICATIONS

Ros, et al., Event-Driven Simulation Scheme for Spiking Neural Networks Using Lookup Tables to Characterize Neuronal Dynamics, Neural Computation 18, 2006, pp. 2959-2993.*

Carrillo, R.R. et al., "Lookup Table Powered Neural Event-Driven Simulator", Computational Intelligence and Bioinspired Systems, Proceedings of the Eighth International Work Conference on Artificial Neural Networks (IWANN '05), 2005, pp. 168-175, Springer-Verlag Berlin, Heidelberg, Germany.

(Continued)

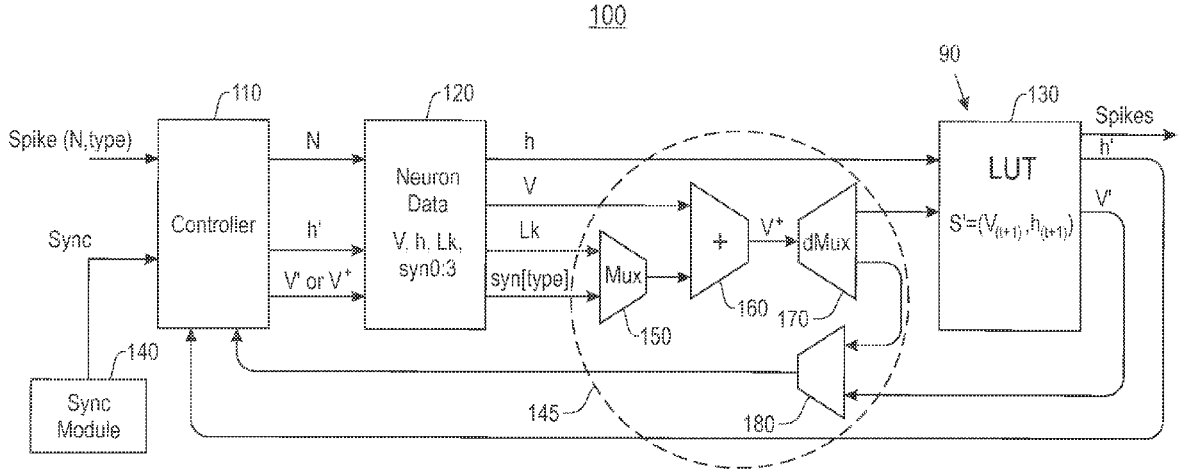
Primary Examiner — Wilbert L Starks

(74) *Attorney, Agent, or Firm* — Sherman IP LLP; Kenneth L. Sherman; Hemavathy Perumal

(57) **ABSTRACT**

Embodiments of the invention relate to a neural network system for simulating neurons of a neural model. One embodiment comprises a memory device that maintains neuronal states for multiple neurons, a lookup table that maintains state transition information for multiple neuronal states, and a controller unit that manages the memory device. The controller unit updates a neuronal state for each neuron based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

27 Claims, 14 Drawing Sheets



(56) **References Cited**

OTHER PUBLICATIONS

Ros, E. et al., "Event-Driven Simulation Scheme for Spiking Neural Networks Using Lookup Tables to Characterize Neuronal Dynamics", Journal of Neural Computation, Dec. 2006, pp. 2959-2993, vol. 18, No. 12, MIT Press, USA.

Izhikevich, E.M., "Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting", Jan. 22, 2010, pp. i-505, MIT Press, USA.

Reutimann, J. et al., "Event-Driven Simulation of Spiking Neurons with Stochastic Dynamics", Journal of Neural Computation, Apr. 2003, pp. 811-830, vol. 15, No. 4, MIT Press, USA.

Kyung, M. et al., "A VLSI Implementation: Izhikevich's Neuron Model", BENG/BGGN 260 Neurodynamics Course: Winter 2009 Project, pp. 1-30. University of California, San Diego, USA.

Alcazar, J.G., "EDLUT: Brief description", edlut: Computer application for simulating networks of spiking neurons, Feb. 4, 2010, pp. 1-3, Google, Inc., USA.

Rochel, O. et al., "An Event-Driven Framework for the Simulation of Networks of Spiking Neurons", Proceedings of the 11th European Symposium on Artificial Neural Networks (ESANN '03), Apr. 23-25, 2003, pp. 295-300, Catholic University of Louvain, Belgium.

Morrison, A. et al., "Advancing the Boundaries of High-Connectivity Network Simulation with Distributed Computing", Journal of Neural Computation, Aug. 2005, pp. 1776-1801, vol. 17, No. 8, MIT Press, USA.

Fiori, S., "Hybrid Independent Component Analysis by Adaptive LUT Activation Function Neurons", Journal of Neural Networks, Jan. 2002, pp. 85-94, vol. 15, No. 1 , Elsevier Science Ltd., United Kingdom.

Muthuramalingam, A. et al., "Neural Network Implementation Using FPGA: Issues and Application", International Journal of Information and Communications Engineering, 2008, pp. 86-92, vol. 4, No. 2, World Academy of Science, Engineering and Technology, USA.

Vogelstein, R.J., et al., "Beyond Address-Event Communication", The Neuromorphic Engineer, 2004, pp. 1-3, Institute of Neuromorphic Engineering, USA.

Zou, Q. et al., "Real-time Simulations of Networks of Hodgkin-Huxley Neurons Using Analog Circuits", Journal of Neurocomputing, Jun. 2006, pp. 1137-1140, vol. 69, No. 10-12, Elsevier Science Publishers B.V., The Netherlands.

Goldberg, D.H. et al., "Probablistic Synaptic Weighting in a Reconfigurable Network of VLSI Integrate-and-Fire Neurons", Journal of Neural Networks, Jul.-Sep. 2001, pp. 781-793, vol. 14, No, 6-7, Elsevier Science Ltd., United Kingdom.

Uncini, A. et al., "Fast Complex Adaptive Spline Neural Networks for Digital Signal Processing", Proceedings of the 1998 IEEE International Joint Conference on Neural Networks (IJCNN '981, May 1998, pp. 903-908, vol. 2, IEEE, USA.

Piazza, F. et al., "Neural Networks with Digital LUT Activation Functions", Proceedings of the 1993 IEEE International Joint Conference on Neural Networks (IJCNN '93), Oct. 1993, pp. 1401-1404, vol. 2, IEEE, USA.

He, M. et al., "Neural Network for Nanoscale Architecture", Proceedings of the Sixth IEEE Conference on Nanotechnology (IEEE-NANO '06), Jun. 2006, pp. 367-370, vol. 1, IEEE, USA.

Vogelstein, R.J. et al., "Dynamically Reconfigurable Silicon Array of Spiking Neurons with Conductance-Based Synapses", IEEE Transactions on Neural Networks, Jan. 2007, pp. 253-265, vol. 18, No. 1, IEEE, USA.

Marchesi, M. et al., "Fast Neural Networks Without Multipliers", IEEE Transactions on Neural Networks, Jan. 1993, pp. 53-62, vol. 4, No. 1, IEEE, USA.

Alvado, L. et al., "Hardward Computation of Conductance-Based Neuron Models", Journal of Neurocomputing, Jun. 2004, pp. 109-115, vol. 58-60, Elsevier B.V., The Netherlands.

Merolla, P. et al., "A Digital Neurosynaptic Core Using Embedded Crossbar Memory with 45pJ per Spike in 45nm", In the Proceedings of the 2011 Custom Integrated Circuits Conference (CICC), Sep. 19-21, 2011, pp. 1-4, IEEE Xplore, United States.

Merolla, P.A. et al., "Expandable Networks for Neuromorphic Chips," IEEE Transactions on Circuits and Systems—I: Regular Papers, Feb. 2007, pp. 301-311, vol. 54, No. 2, IEEE, United States.

U.S. Non-Final Office Action for U.S. Appl. No. 13/725,442 mailed Oct. 16, 2014.

U.S. Notice of Allowance for U.S. Appl. No. 13/725,442 mailed Jan. 30, 2015 by Examiner Peter D. Coughlan.

* cited by examiner

1

$$dV/dt = f(V, h)$$

$$dh/dt = g(V, h)$$

$$\left.\begin{aligned} V &= V + dV*dt \\ h &= h + dh*dt \end{aligned}\right\} s$$

FIG. 1

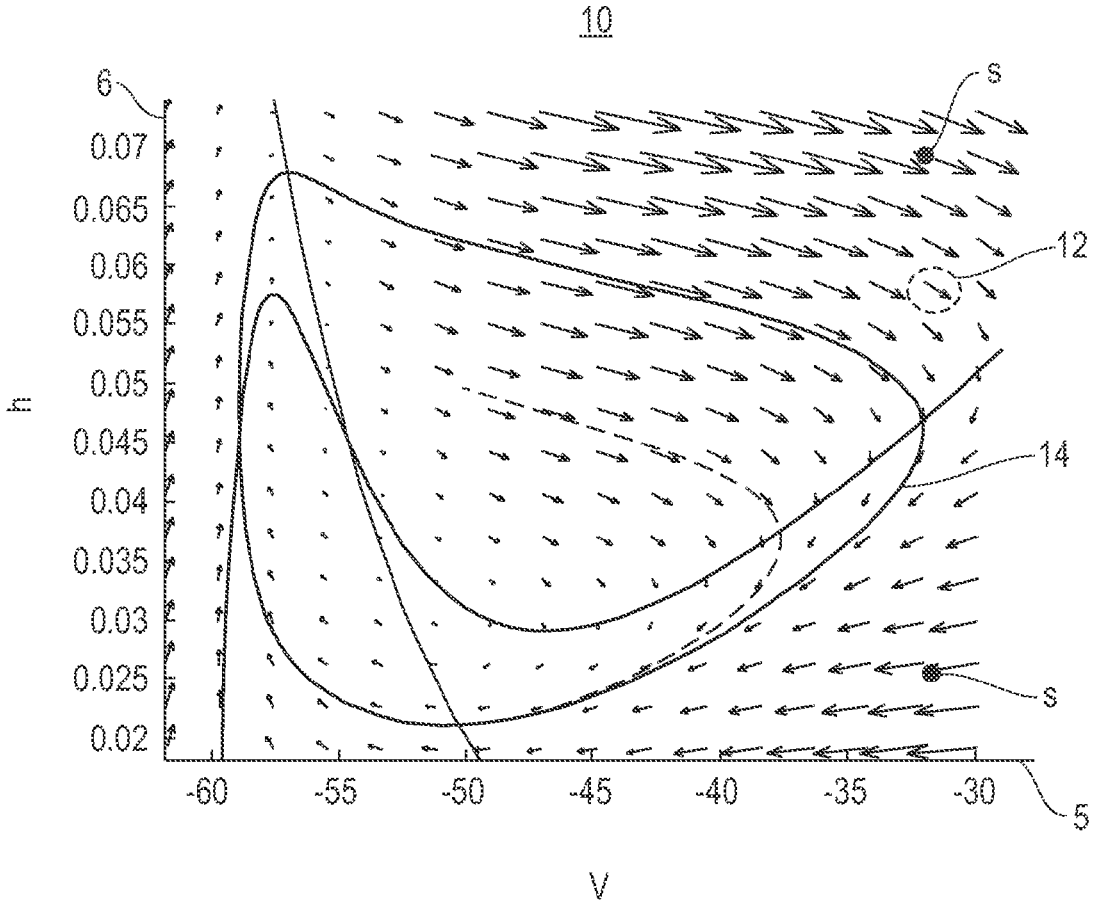


FIG. 2A

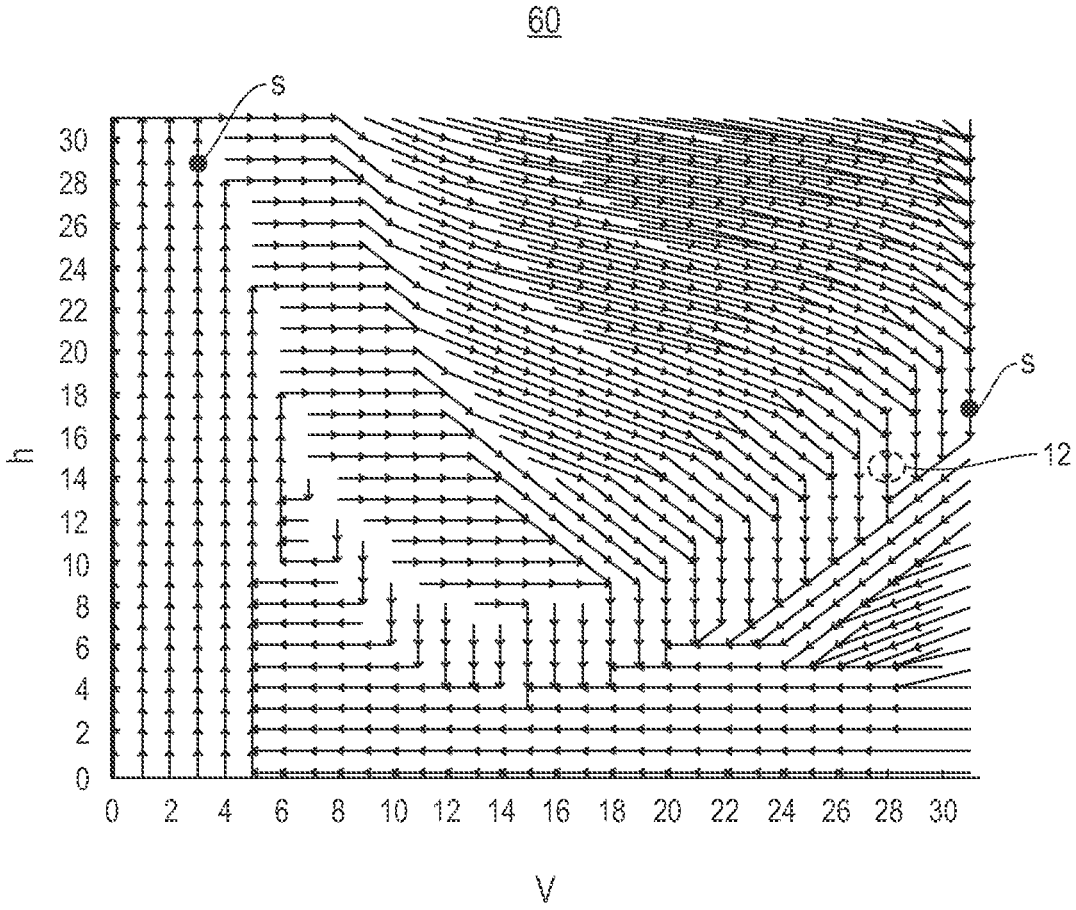


FIG. 2B

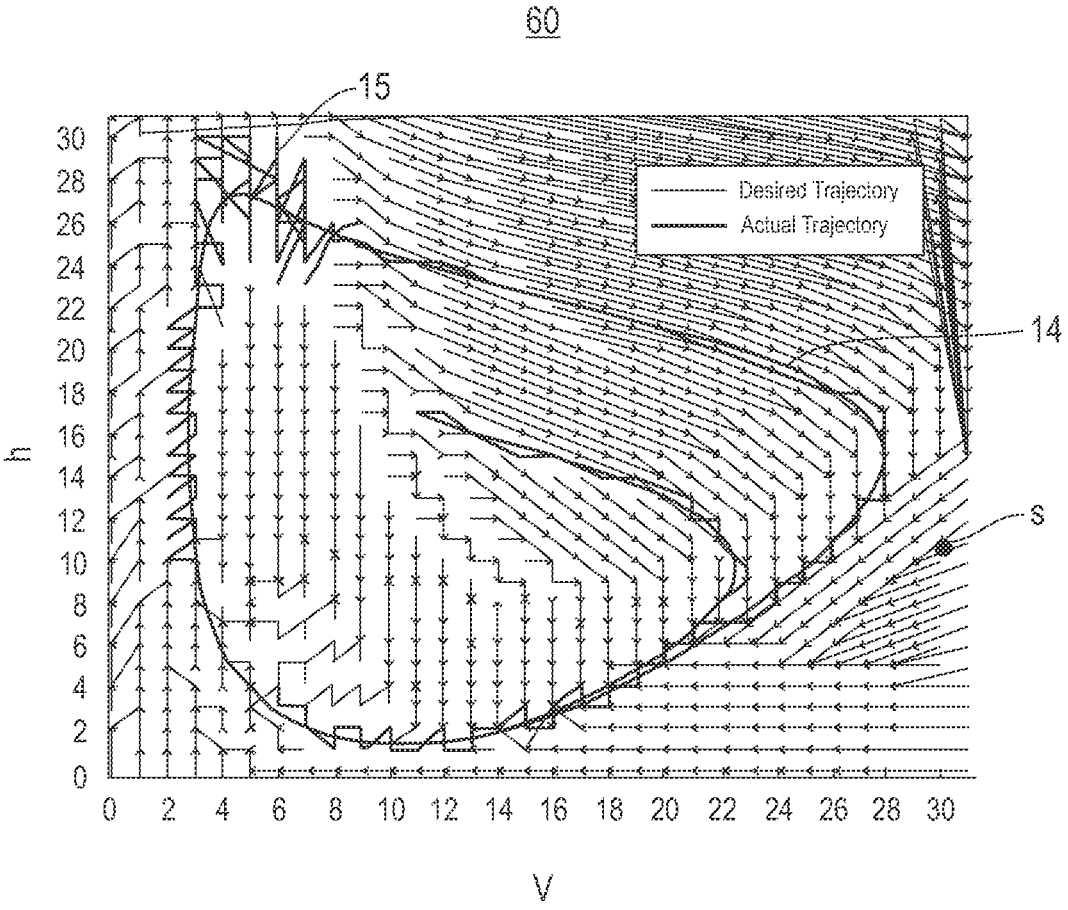

60
15
Desired Trajectory
Actual Trajectory
14
s
h
V
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30

FIG. 3

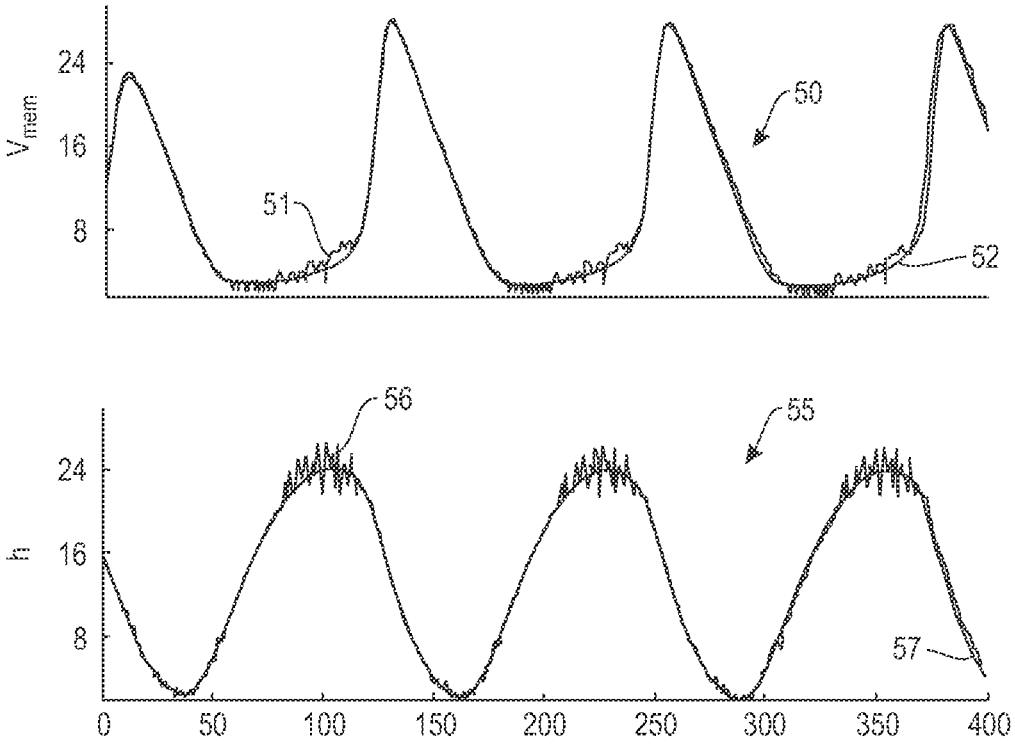

V_{mem}
24
16
8
50
51
52
56
55
57
I_m
24
16
8
0
50
100
150
200
250
300
350
400

FIG. 4

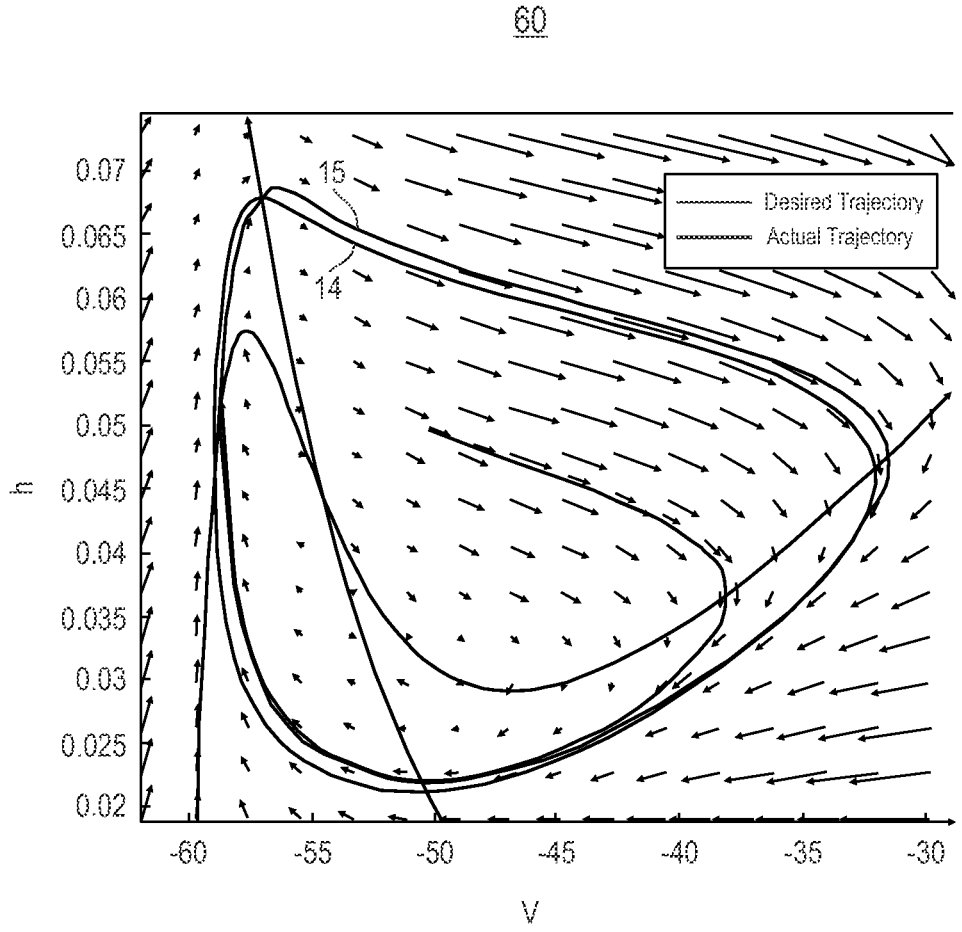

60
15
14
Desired Trajectory
Actual Trajectory
0.07
0.065
0.06
0.055
0.05
0.045
0.04
0.035
0.03
0.025
0.02
n
-60
-55
-50
-45
-40
-35
-30
V

FIG. 5

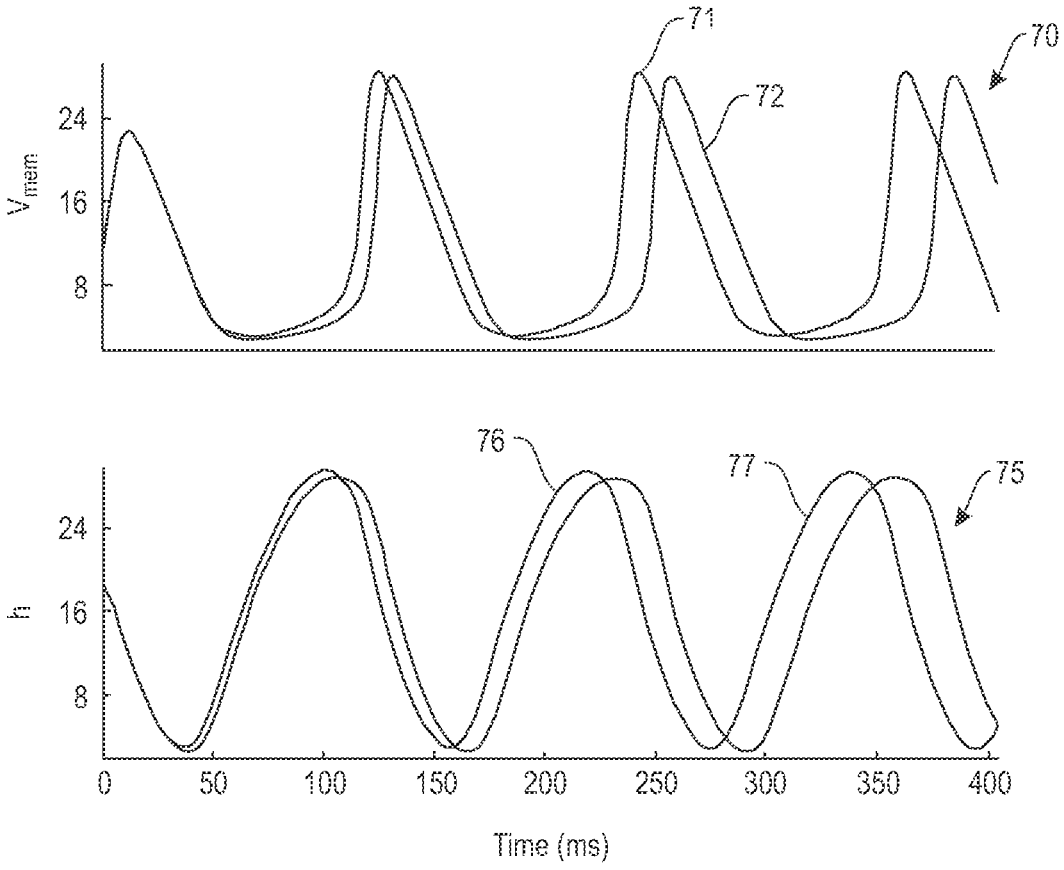

71
70
72
24
16
8
V_{mem}
76
77
75
24
16
8
I_h
0
50
100
150
200
250
300
350
400
Time (ms)

FIG. 6

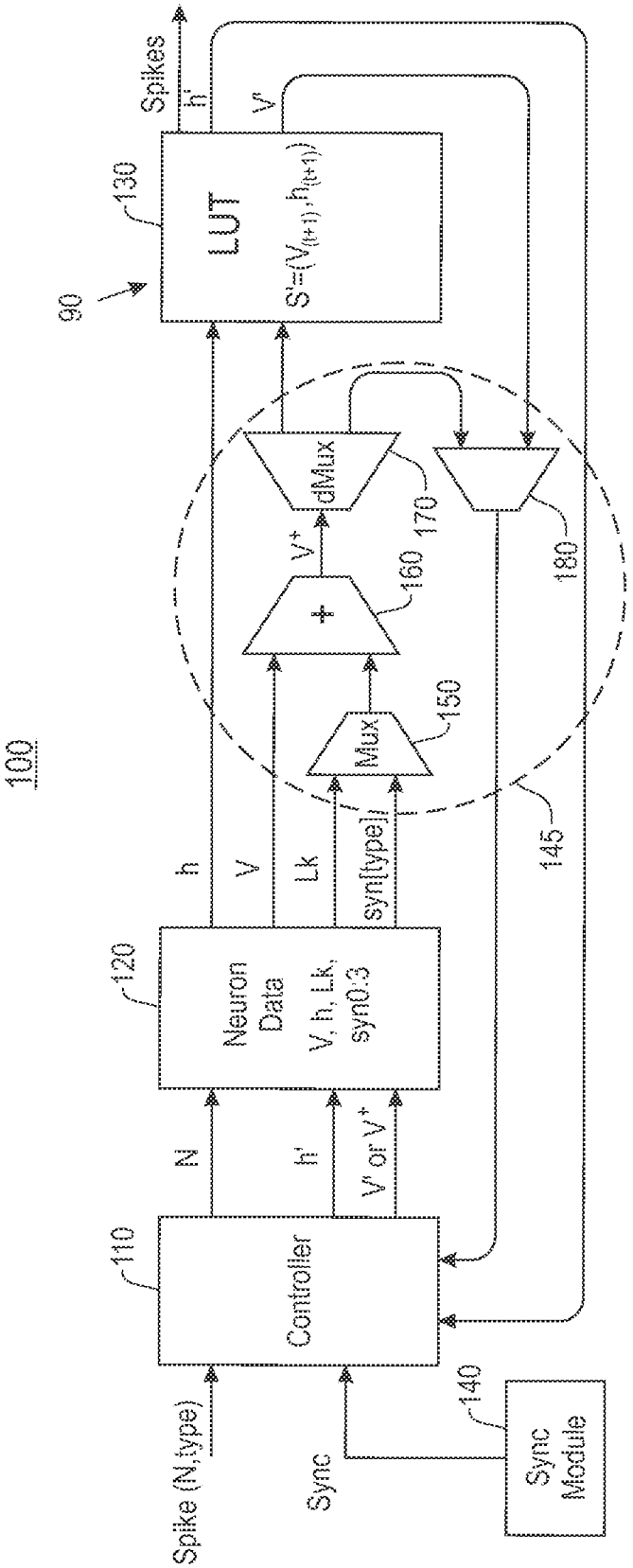

100
Spike (N,type)
Sync
Sync Module
140
Controller
110
N
h'
V' or V+
Neuron Data V, h, Lk, syn0:3
120
h
V
Lk
syn[type]
Mux
150
+
160
V+
dMux
170
180
145
90
LUT
S'=(V(t+1), h(t+1))
130
Spikes
h'
V'

FIG. 7

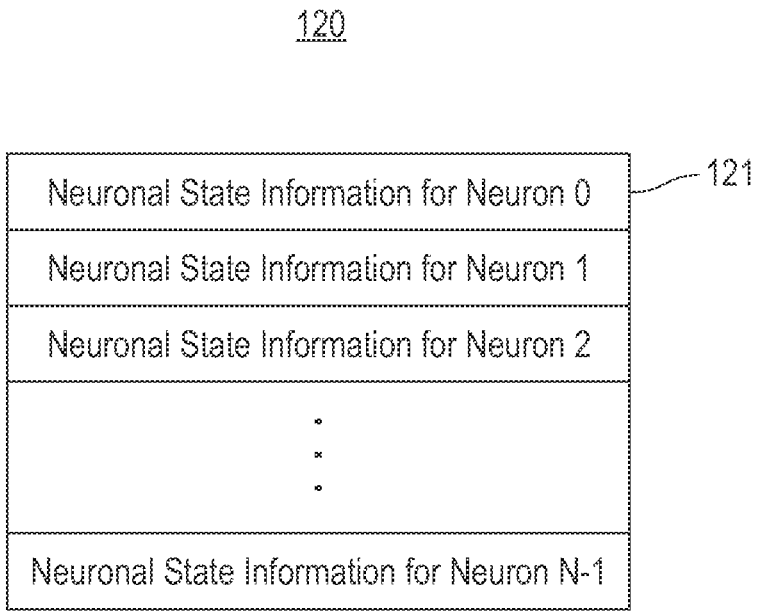

120
Neuronal State Information for Neuron 0
121
Neuronal State Information for Neuron 1
Neuronal State Information for Neuron 2
Neuronal State Information for Neuron N-1

FIG. 8

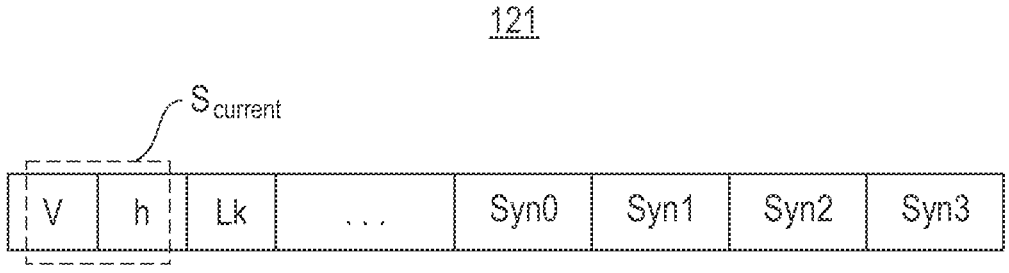

121
S current
V
h
Lk
. . .
Syn0
Syn1
Syn2
Syn3

FIG. 9

130

Current State	Next State
s (V,h)	$s_{(t+1)}(V_{(t+1)},h_{(t+1)})$
0 (0,0)	64 (0,2)
1 (1,0)	33 (1,1)
2 (2,0)	34 (2,1)
⋮	⋮
1021 (29,31)	991 (31,30)
1022 (30,31)	991 (31,30)
1023 (31,31)	991 (31,30)

s

S_{next}

131

FIG. 10

131

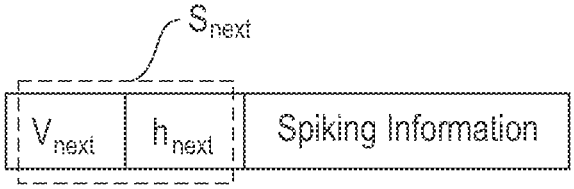


FIG. 11

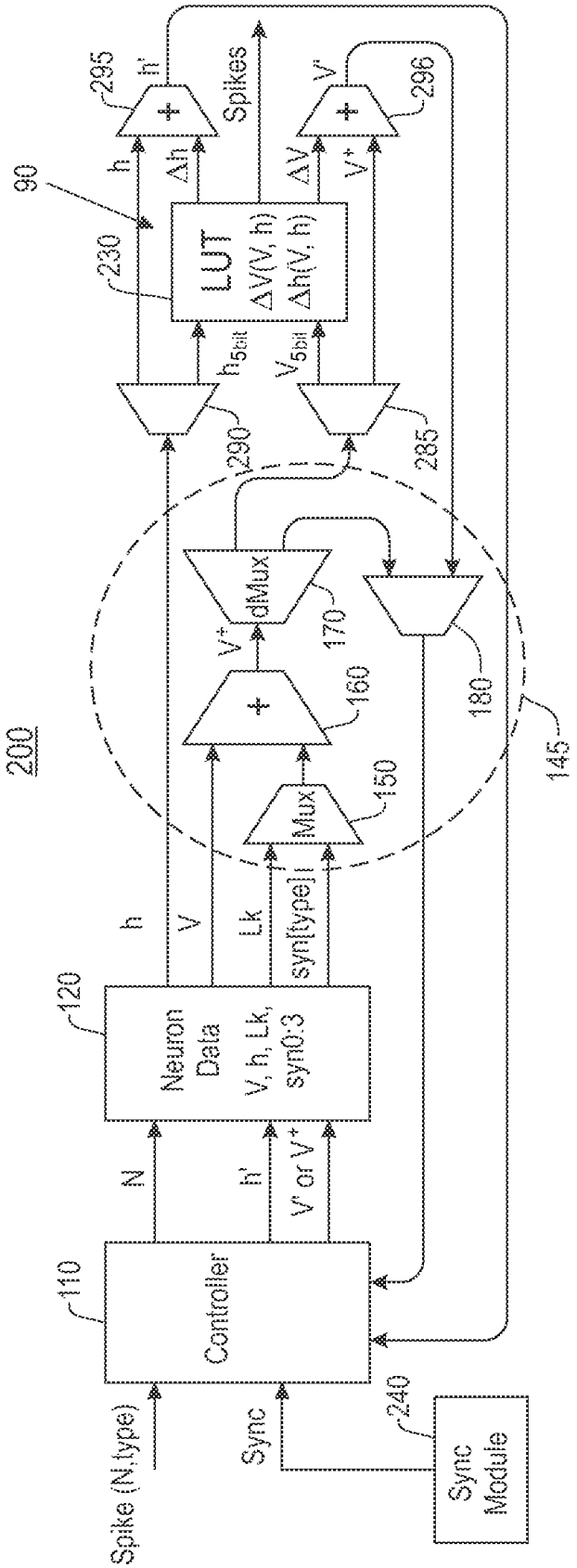

200
90
Spike (N,type)
Sync
240
Sync Module
110
Controller
N
h'
V' or V+
120
Neuron Data V, h, Lk, syn0:3
h
V
Lk
syn[type]
150
Mux
160
+
V+
170
dMux
180
145
290
285
230
LUT ΔV(V, h) Δh(V, h)
h5bit
V5bit
h
Δh
ΔV
V+
295
h'
296
V'
Spikes

FIG. 12

230

Current State	Next State	
s (V, h)	ΔV (V, h)	Δh (V, h)
0 (0,0)	0	2
1 (1,0)	0	1
2 (2,0)	0	1
⋮		

s

231

FIG. 13

231

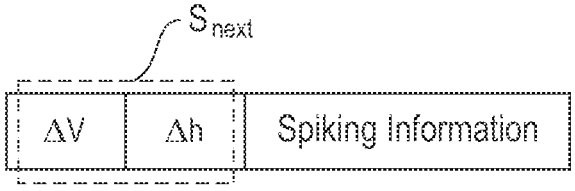


FIG. 14

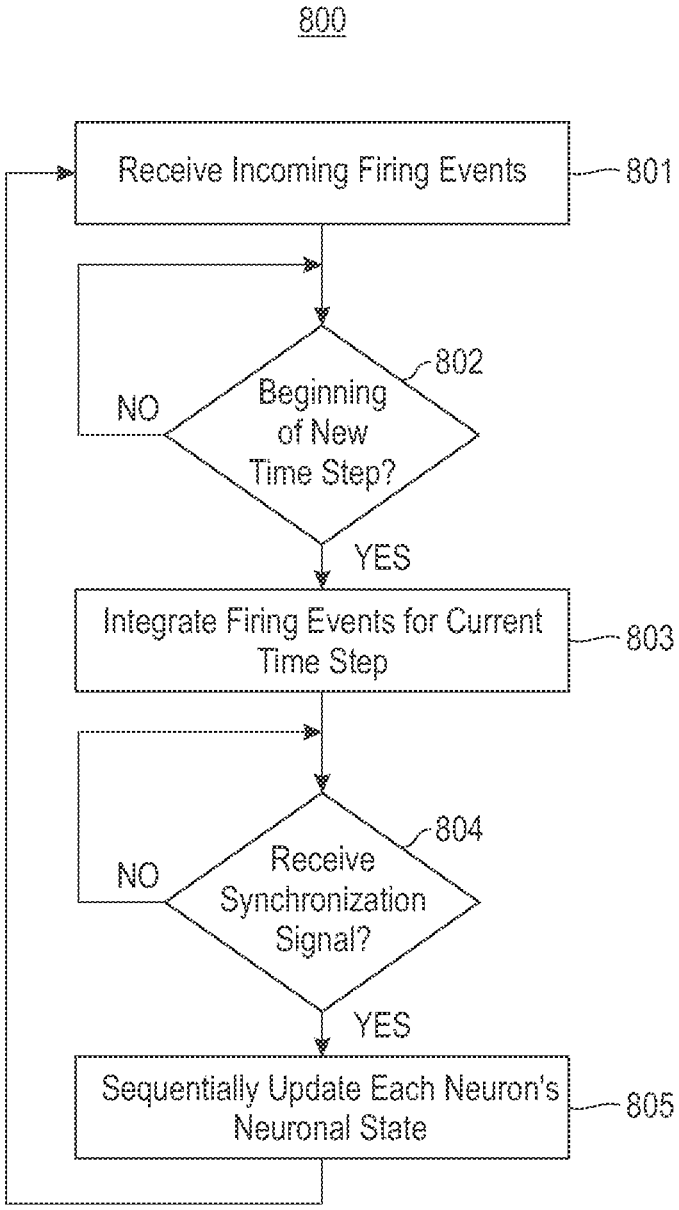

800
Receive Incoming Firing Events
801
802
Beginning of New Time Step?
NO
YES
Integrate Firing Events for Current Time Step
803
804
Receive Synchronization Signal?
NO
YES
Sequentially Update Each Neuron's Neuronal State
805

FIG. 15

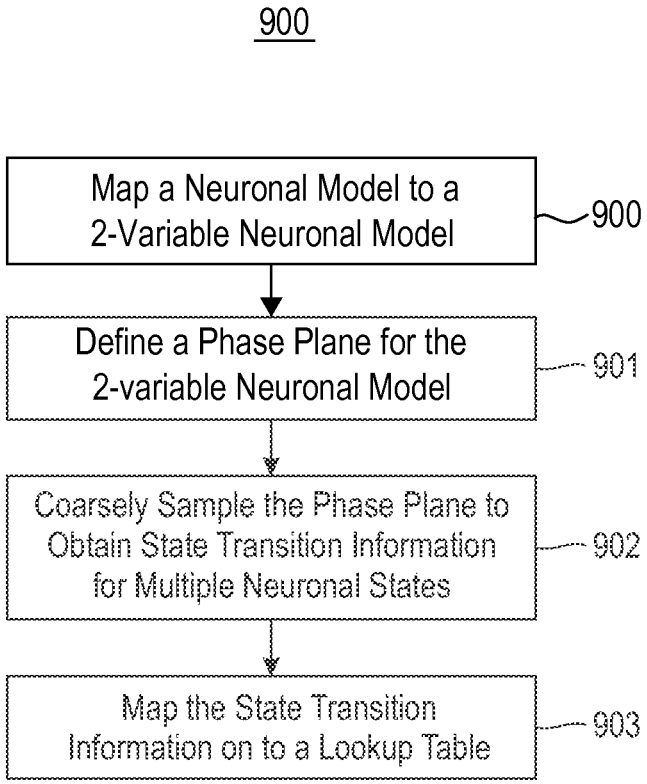

900
Map a Neuronal Model to a 2-Variable Neuronal Model
900
Define a Phase Plane for the 2-variable Neuronal Model
901
Coarsely Sample the Phase Plane to Obtain State Transition Information for Multiple Neuronal States
902
Map the State Transition Information on to a Lookup Table
903

FIG. 16

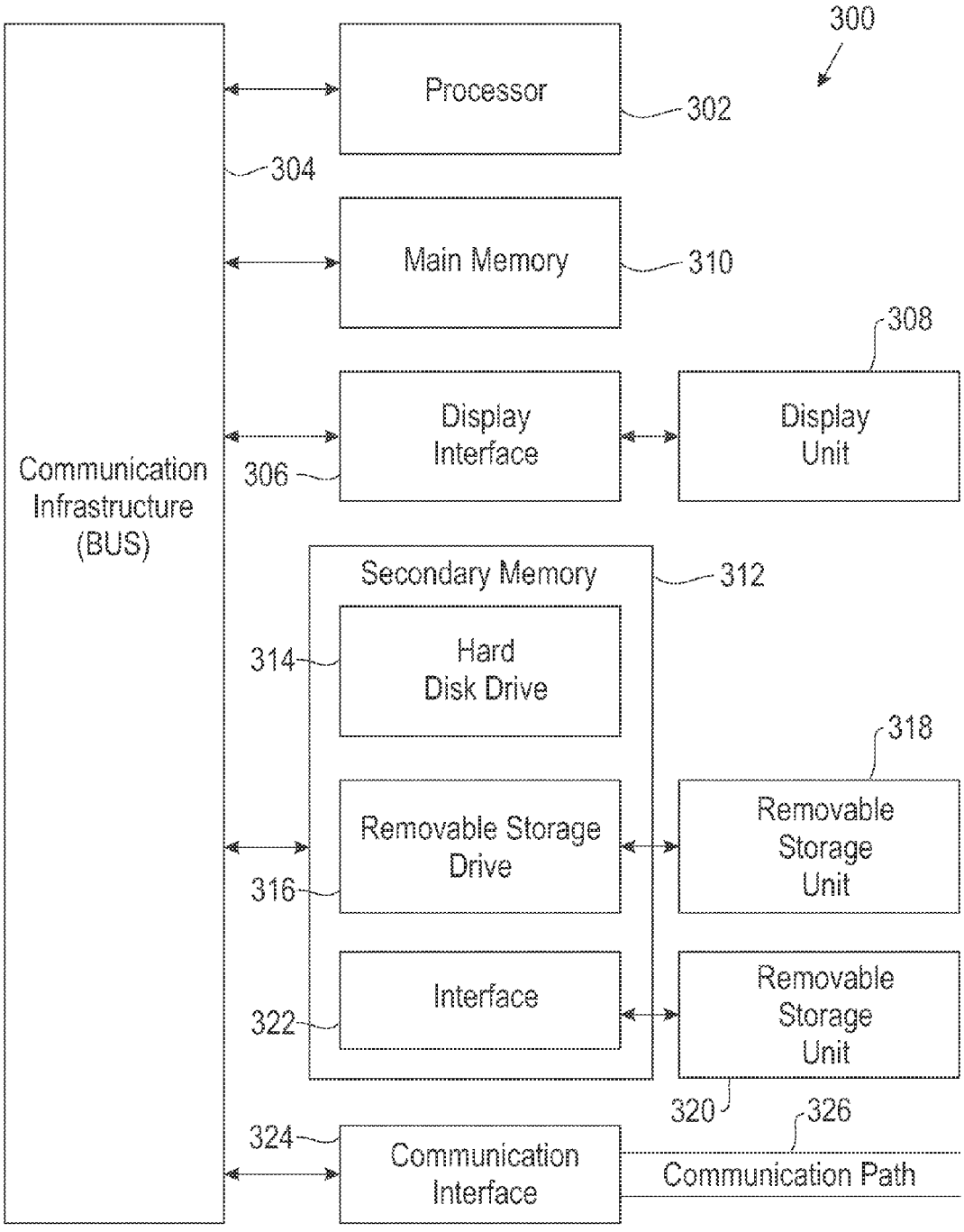

300
Processor
302
304
Main Memory
310
308
Display Interface
Display Unit
306
Communication Infrastructure (BUS)
Secondary Memory
312
314
Hard Disk Drive
318
Removable Storage Drive
Removable Storage Unit
316
Interface
Removable Storage Unit
322
320
326
324
Communication Interface
Communication Path

FIG. 17

HARDWARE ARCHITECTURE FOR SIMULATING A NEURAL NETWORK OF NEURONS

This invention was made with Government support under HR0011-09-C-0002 awarded by Defense Advanced Research Projects Agency (DARPA). The Government has certain rights in this invention.

BACKGROUND

Embodiments of the invention relate to neuromorphic and synaptronic computation and in particular, a neural network system for simulating neurons of a neural model.

Neuromorphic and synaptronic computation, also referred to as artificial neural networks, are computational systems that permit electronic systems to essentially function in a manner analogous to that of biological brains. Neuromorphic and synaptronic computation do not generally utilize the traditional digital model of manipulating 0s and 1s. Instead, neuromorphic and synaptronic computation create connections between processing elements that are roughly functionally equivalent to neurons of a biological brain. Neuromorphic and synaptronic computation may comprise various electronic circuits that are modeled on biological neurons.

In biological systems, the point of contact between an axon of a neuron and a dendrite on another neuron is called a synapse, and with respect to the synapse, the two neurons are respectively called pre-synaptic and post-synaptic. The essence of our individual experiences is stored in conductance of the synapses. The synaptic conductance changes with time as a function of the relative spike times of pre-synaptic and post-synaptic neurons, as per spike-timing dependent plasticity (STDP). The STDP rule increases the conductance of a synapse if its post-synaptic neuron fires after its pre-synaptic neuron fires, and decreases the conductance of a synapse if the order of the two firings is reversed.

BRIEF SUMMARY

Embodiments of the invention relate to a neural network system for simulating neurons of a neural model. One embodiment comprises a memory device that maintains neuronal states for multiple neurons, a lookup table that maintains state transition information for multiple neuronal states, and a controller unit that manages the memory device. The controller unit updates a neuronal state for each neuron based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

Another embodiment comprises an apparatus including a memory device and a controller unit. The memory device includes a lookup table that maintains information about multiple system states of a model system, wherein the lookup table is configured to provide information relating to a change in a system state when addressed by a current system state of the model system. The controller unit is configured to update system states for multiple instances of the model system based on the information maintained in lookup table and incoming data received by the controller unit.

Another embodiment comprises maintaining neuronal states for multiple neurons, and maintaining state transition information for multiple neuronal states in a lookup table. A neuronal state for each neuron is updated based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

Another embodiment comprises maintaining system states for multiple instances of a model system, and maintaining system state change information for the system states in a lookup table. The system state change information comprises information relating to changes in said system states. For each instance of the model system, the system state for said instance is updated based on incoming data and the system state change information maintained.

These and other features, aspects, and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

FIG. **1** illustrates an example neural model, in accordance with an embodiment of the invention;

FIG. **2**A illustrates an example phase plane for the neural model in FIG. **1**, in accordance with an embodiment of the invention;

FIG. **2**B illustrates an example discretized phase plane of the phase plane in FIG. **2**A, in accordance with an embodiment of the invention;

FIG. **3** illustrates an example discretized phase plane for a neural model, wherein the discretized phase plane includes a desired trajectory computed based on the neural model and a resulting actual trajectory computed based on the discretized phase plane, in accordance with an embodiment of the invention;

FIG. **4** illustrates waveforms for the example discretized phase plane in FIG. **3**, in accordance with an embodiment of the invention;

FIG. **5** illustrates an example coarsely discretized phase plane for a neural model, in accordance with an embodiment of the invention;

FIG. **6** illustrates graphs for the example discretized phase plane in FIG. **5**, in accordance with an embodiment of the invention;

FIG. **7** illustrates an example neural network system for simulating neurons of a neural model, wherein the system includes a lookup table that maintains a sequence of neuronal states, in accordance with an embodiment of the invention;

FIG. **8** illustrates example neuronal state information maintained in the memory device in FIG. **7**, in accordance with an embodiment of the invention;

FIG. **9** illustrates example neuronal state information maintained in an entry of the memory device in FIG. **8**, in accordance with an embodiment of the invention;

FIG. **10** illustrates example state transition information maintained in the lookup table in FIG. **7**, in accordance with an embodiment of the invention;

FIG. **11** illustrates example state transition information maintained in an entry of the lookup table in FIG. **10**, in accordance with an embodiment of the invention;

FIG. **12** illustrates an example neural network system for simulating neurons of a neural model, wherein the system includes a delta-based lookup table that maintains high-resolution delta values of at least one variable of the neural model, in accordance with an embodiment of the invention;

FIG. **13** illustrates example state transition information maintained in the lookup table in FIG. **12**, in accordance with an embodiment of the invention;

FIG. **14** illustrates example state transition information maintained in an entry of the lookup table in FIG. **13**, in accordance with an embodiment of the invention;

FIG. **15** illustrates a flowchart of an example process for integrating synaptic firing events and updating each neuron's neuronal state, in accordance with an embodiment of the invention;

FIG. **16** illustrates a flowchart of an example process for mapping neural dynamics of a neural model on to a lookup table, in accordance with an embodiment of the invention; and

FIG. **17** is a high-level block diagram showing an information processing system useful for implementing one embodiment of the invention.

DETAILED DESCRIPTION

Embodiments of the invention relate to a neural network system for simulating neurons of a neural model. One embodiment comprises a memory device that maintains neuronal states for multiple neurons, a lookup table that maintains state transition information for multiple neuronal states, and a controller unit that manages the memory device. The controller unit updates a neuronal state for each neuron based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

Another embodiment of the invention comprises maintaining neuronal states for multiple neurons, and maintaining state transition information for multiple neuronal states in a lookup table. A neuronal state for each neuron is updated based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

The neurons are based on a neural model. For each time step, the controller unit updates a neuronal state for each neuron. A neuronal state transitions to a subsequent neuronal state in response to changes in at least one variable of the neural model.

The memory device has multiple entries, wherein each entry maintains a neuronal state for a corresponding neuron. The lookup table is addressed by a neuronal state. The lookup table has multiple entries, wherein each entry corresponds to a neuronal state. Each entry maintains state transition information for a corresponding neuronal state. In one embodiment, state transition information for a neuronal state identifies a subsequent neuronal state that said neuronal state transitions to. In another embodiment, state transition information for a neuronal state comprises at least one high-resolution delta value corresponding to a variable of the neural model, wherein each delta value is added to said neuronal state to obtain a subsequent neuronal state that said neuronal state transitions to.

A computation circuit integrates each incoming spike event received. In one embodiment, the lookup table further comprises spiking information for multiple neuronal states.

A neuronal state includes at least one variable of the neural model. The lookup table is addressed by combining two or more variables of the neural model.

One embodiment comprises an apparatus including a memory device and a controller unit. The memory device includes a lookup table that maintains information about multiple system states of a model system, wherein the lookup table is configured to provide information relating to a change in a system state when addressed by a current system state of the model system. The controller unit is configured to update system states for multiple instances of the model system based on the information maintained in lookup table and incoming data received by the controller unit.

Another embodiment comprises maintaining system states for multiple instances of a model system, and maintaining system state change information for the system states in a lookup table. The system state change information comprises information relating to changes in said system states. For each instance of the model system, the system state for said instance is updated based on incoming data and the system state change information maintained.

In one embodiment, for each system state in the lookup table, the system state is represented as an n-bit digital number in the lookup table, wherein n is a natural number. Information relating to a change in the system state is represented as an m-bit digital number in the lookup table, wherein m is a natural number. For the current system state of the model system, the controller unit is further configured to add incoming data and an m-bit digital number representing a change in the current system state to an n-bit digital number representing the current system state to determine a next system state for the current system state. The controller unit is further configured to add incoming data and a subset of the m-bit digital number representing a change in the current system state to a subset of the n-bit digital number representing the current system state to determine a next system state for the current system state.

In another embodiment, for each system state in the lookup table, the system state is represented as an n-bit digital number in the lookup table, wherein n is a natural number. A next system state for the system state is represented as an m-bit digital number in the lookup table, wherein m is a natural number. For the current system state of the model system, the controller unit is further configured to combine incoming data and an n-bit digital number representing the current system state to determine a next system state for the current system state.

In one embodiment, the model system is a neural system. For each system state in the lookup table, the lookup table further includes additional information about whether the model system is spiking at the system state. In another embodiment, for each system state in the lookup table, the lookup table further includes additional metadata information about the system state.

In one embodiment, the memory device and the controller unit are on the same integrated circuit. In another embodiment, the memory device and the controller unit are on different integrated circuits. The different integrated circuits are interconnected.

The term digital neuron as used herein represents a framework configured to simulate or emulate a biological neuron. A digital neuron creates connections between processing elements that are roughly functionally equivalent to neurons of a biological brain. As such, a neuromorphic and synaptronic computation comprising digital neurons, according to embodiments of the invention, may include various electronic circuits that are modeled on biological neurons. Further, a neuromorphic and synaptronic computation comprising digital neurons, according to embodiments of the invention, may include various processing elements (including computer simulations) that are modeled on biological neurons. Although certain illustrative embodiments of the invention are described herein using digital neurons comprising digital circuits, the present invention is not limited to digital circuits. A neuromorphic and synaptronic computation, according to embodiments of the invention, can be implemented as a neuromorphic and synaptronic framework comprising circuitry and additionally as a computer simulation. Indeed, embodiments of the invention can take the form of an entirely hardware embodiment, an entirely software embodiment, or an embodiment containing both hardware and software elements.

FIG. **1** illustrates an example neural model 1, in accordance with an embodiment of the invention. A digital neuron may be modeled on a neural model 1. In this specification, the progression of a neural model 1 over time is referred to as temporal neural dynamics. Neural dynamics of a neural model 1 may be expressed using one or more variables. There are several different types of neural models **1**, such as an integrate and fire neural model, a resonate and fire neural model, an Izhikevich neural model, etc.

In one embodiment, a 2-variable neural model 1 with a first variable V and a second variable h may be expressed using the following equations: (1) dV/dt=f(V, h), wherein dV/dt represents the rate of change of the first variable V over time; (2) dh/dt=g(V, h), wherein dh/dt represents the rate of change of the second variable h over time; (3) V=V+dV*dt; and (4) h=h+dh*dt. f(V, h) and g(V, h) may be any function that represents a type of neural model 1. The first variable V and the second variable h represent a neuronal state S of a neuron. In one embodiment, the first variable V represents a membrane potential variable, and the second variable h represents an internal state variable (e.g., an ion-channel's activation or inactivation).

FIG. **2**A illustrates an example phase plane **10** for the neural model 1 in FIG. **1**, in accordance with an embodiment of the invention. A phase plane **10** represents a state space of a neural model 1 (FIG. **1**) and how a neuron's state evolves over time. A phase plane **10** is computed by sampling the equations representing the neural model 1.

As shown in FIG. **2**A, a horizontal axis **5** and a vertical axis **6** of the phase plane **10** are denoted as V and h, respectively. The phase plane **10** includes at least one arrow **12**. Each arrow **12** represents a state transition from a neuronal state S to another neuronal state S. Each arrow **12** is generated by evaluating f(V, h) and g(V, h) at a coordinate of the phase plane **10**.

FIG. **2**A also shows a computed desired trajectory **14** of the neural model 1. Table 1 below provides example pseudo code that may be used to calculate the desired trajectory **14**.

TABLE 1

```
// Seed initial state
V = V_initial;
h = h_initial;
// Execute the following code in a loop
V_next = V + f(V,h)*dt;
h_next = h + g(V,h)*dt;
V = V_next;
h = h_next;
```

FIG. **2**B illustrates an example discretized phase plane **60** of the phase plane **10** in FIG. **2**A, wherein each state transition **12** is scaled to point to another sampling point, in accordance with an embodiment of the invention. The neural dynamics of a neural model 1 may be mapped on to a lookup table **90** (FIG. **7**) by sampling a phase plane **10** representing the state space of the neural model 1. Sampling the phase plane **10** comprises obtaining a finite number of neuronal states S.

In one embodiment, the phase plane **10** is discretized into a finite number of neuronal states S. For example, the phase plane **10** may be discretized to generate a discretized phase plane **60**. Each arrow **12** in FIG. **2**B represents a state transition from one neuronal state S (e.g., a neuronal state $S_{current}$ at the current time step t) to another state s (e.g., a next neuronal state S_{next} at the next time step t+1). The discretized phase plane **60** is sampled to map information relating to state transitions on to a lookup table **90**.

There are multiple ways of sampling a phase plane **10**. In one embodiment, the equations representing a neural model 1 (e.g., f(V, h) and g(V, h)) are evaluated at different sampling points of the discretized phase plane **60**, wherein each sampling point represents an neuronal state S. Each result is rounded to the nearest sampling point of the discretized phase plane **60**.

FIG. **3** illustrates an example discretized phase plane **60** for a neural model 1, wherein the discretized phase plane **60** includes a desired trajectory **14** computed based on the neural model and a resulting actual trajectory computed based on the discretized phase plane, in accordance with an embodiment of the invention. In one embodiment, a desired trajectory **14** representing the progression of a neural model 1 over time is computed using the equations representing the neural model 1 (e.g., f(V, h) and g(V, h)). Sampling points representing different neuronal states S are discretized along, or within the proximity of, the computed desired trajectory **14** to generate an actual trajectory **15**. Noise may be added to areas of the discretized phase plane **60** where neural dynamics are slower. Adding noise prevents the actual trajectory **15** from entering a dead zone.

As shown in FIG. **3**, despite the added noise, the actual trajectory **15** is still as close as possible to the computed desired trajectory **14**.

In one embodiment, mapping a two-state neural model 1 to a lookup table comprises selecting maximum and minimum values for V and h (V_{max}, V_{min}, h_{max}, and h_{min}). The maximum and minimum values for V and h are divided by the lookup table's number of states for V and h to determine the size of the discretized steps. For example, $V_{step}=(V_{max}-V_{min})/2\hat{}V_{bits}$, and $h_{step}=(h_{max}-h_{min})/2\hat{}h_{bits}$), wherein V_{bits} and h_{bits} represent the number of bits to store V and h, respectively.

A computed trajectory is then mapped. For example, a transient simulation is computed (e.g., using Euler's method) to determine a trajectory within the discretized phase plane **60**. For each timestep of the transient simulation, a neuronal state is rounded to the closest sampling point within the discretized phase plane **60**. For example, $p(t)=\text{round}((V(t)-V_{min})/V_{step})$; $q(t)=\text{round}((h(t)-h_{min})/h_{step})$.

Dead zones (i.e., zones where a neuronal state S along the computed trajectory points to itself) within the discretized phase plane **60** are avoided by checking for duplicated neuronal states in adjacent timesteps. The dead zones are eliminated by offsetting duplicated neuronal states to nearby neuronal states. For example, for each duplicated neuronal state, potential alternate neuronal states are created around a sampling point in the discretized phase plane **60**. Alternate neuronal states that are already occupied within some $\pm\Delta t$ of the current timestep t are discarded. The duplicated neuronal states are replaced with alternate neuronal states that are vacant and closest in distance to the original neuronal state within the discretized phase plane **60**. The trajectory is then mapped to a lookup table. Similarly, the entire discretized phase plane **60** is mapped by computing a trajectory from every sampled neuronal state within the discretized phase plane **60**. All neuronal states are constrained within the discretized phase plane **60**.

FIG. **4** illustrates waveforms for the example discretized phase plane **60** in FIG. **3**, in accordance with an embodiment of the invention. A first graph **50** represents the first variable V with respect to time. The first graph **50** shows a first waveform **51** and a second waveform **52** corresponding to the actual trajectory **15** and the desired trajectory **14** in FIG. **3**, respectively. As shown in FIG. **4**, the first waveform **51** substantially matches the second waveform **52**.

Also shown in FIG. **4** is a second graph **55** that represents the second variable h with respect to time. The second graph **55** shows a third waveform **56** and a fourth waveform **57** corresponding to the actual trajectory **15** and the desired trajectory **14** in FIG. **3**, respectively. As shown in FIG. **4**, the third waveform **56** substantially matches the fourth waveform **57**.

FIG. **5** illustrates an example coarsely discretized phase plane **60** for a neural model 1, in accordance with an embodiment of the invention. In one embodiment, the phase plane **10** (FIG. **2**) is coarsely discretized. To precisely capture the neural model 1's slowest and fastest neural dynamics, high-resolution delta values for variables of the neural model 1, such as V and h, are maintained in a delta-based lookup table **230** (FIG. **12**). Maintaining high-resolution delta values generates the actual trajectory **15**, wherein the actual trajectory **15** is smooth and qualitatively similar to a computed desired trajectory **14**.

In one embodiment, mapping a two-state neural model 1 to a delta-based lookup table comprises selecting maximum and minimum values for V and h (V_{max}, V_{min}, h_{max}, and h_{min}). The maximum and minimum values for V and h are divided by the delta-based lookup table's number of states for V and h to determine the size of the discretized steps. For example, $V_{step}=(V_{max}-V_{min})/2\hat{}V_{bits}$, and $h_{step}=(h_{max}-h_{min})/2\hat{}h_{bits}$), wherein V_{bits} and h_{bits} represent the number of bits to store V and h, respectively.

dV=f(V,h)*dt and dh=g(V,h)*dt are computed for each sample point within the discretized phase plane **60**. dV and dh are rounded up and discretized into binary ΔV and Δh that will populate the delta-based lookup table. Stable equilibrium points are found, and dV and dh are updated for all equilibrium points. The dV and dh values are constrained such that a neuronal state is constrained within the discretized phase plane **60**. All binary ΔV and Δh values are then mapped to a delta-based lookup table.

FIG. **6** illustrates waveforms for the example discretized phase plane **60** in FIG. **5**, in accordance with an embodiment of the invention. A first graph **70** represents the first variable V with respect to time. The first graph **70** shows a first waveform **71** and a second waveform **72** corresponding to the actual trajectory **15** and the desired trajectory **14** in FIG. **5**, respectively. As shown in FIG. **6**, the first waveform **71** is smooth and qualitatively similar to the second waveform **72**. The difference in periods between the first waveform **71** and the second waveform **72** is attributable to discretization error.

Also shown in FIG. **6** is a second graph **75** that represents the second variable h with respect to time. The second graph **75** shows a third waveform **76** and a fourth waveform **77** corresponding to the actual trajectory **15** and the desired trajectory **14** in FIG. **5**, respectively. As shown in FIG. **6**, the third waveform **76** is smooth and qualitatively similar to the fourth waveform **77**. The difference in periods between the third waveform **76** and the fourth waveform **77** is attributable to discretization error.

FIG. **7** illustrates an example neural network system **100** for simulating neurons of a neural model 1, wherein the system **100** includes a lookup table **130** that maintains a sequence of neuronal states S, in accordance with an embodiment of the invention. The system **100** comprises a controller unit **110**, a memory device **120**, a lookup table **130**, a computation fabric **145**, and a sync module **140**.

The memory device **120** is a memory array that maintains neuronal state information for multiple neurons of the same neural model 1 (i.e., neurons of the same type). The neuronal state information for a neuron includes the current neuronal state $S_{current}$ of the neuron (i.e., the neuronal state of the neuron at the current time step). A neuronal state S of a neuron may be expressed using one or more variables of the neural model 1, such as a membrane potential variable V and an internal state variable h. The neuronal state information for a neuron may further include neuron parameters, such as a leak rate Lk and synaptic weights for different types of synaptic connections (e.g., Syn0, Syn1, Syn2, Syn3).

The controller unit **110** manages all synaptic inputs received. In one embodiment, the controller unit **110** receives incoming spike events targeting neurons of the system **100**. Each incoming spike event is tagged with information identifying a target neuron for said spike event, and a type of synaptic connection that said spike event was received from.

The computation fabric **145** integrates each incoming spike event received.

The sync module **140** generates synchronization signals for the controller unit **110**. The controller unit **110** sequentially updates the current neuronal state $S_{current}$ of each neuron of the system **100** when the controller unit **110** receives a synchronization signal.

The lookup table **130** maintains state transition information for a finite number of neuronal states S. Specifically, the lookup table **130** maintains a sequence of neuronal states S. The state transition information for each neuronal state S identifies a next neuronal state S_{next} (i.e., a subsequent neuronal state) that said state s transitions to. A neuronal state S is used to address (i.e., index) the lookup table **130**. For example, the lookup table **130** is addressed by combining the most significant bits of the membrane potential variable V with the most significant bits of the internal state variable h.

In one embodiment, for each incoming spike event received in a current time step t, the controller unit **110** sends a read request to the memory device **120** for the following neuron parameters: the membrane potential variable V of a target neuron in the current time step t, and the synaptic weight for the type of synaptic connection that said spike event was received from. The computation fabric **145** generates a modified membrane potential variable V^+ for the target neuron by adding the synaptic weight to the membrane potential variable V (i.e., integrating the spike event). The controller unit **110** writes the modified membrane potential variable V' to the memory device **120**.

The memory device **120** is read out sequentially when the controller unit **110** receives a synchronization signal. For each neuron of the system **100**, the controller unit **110** sends a read request to the memory device **120** for the membrane potential variable V of the neuron, the internal state variable h of the neuron, and the leak rate Lk of the neuron. The computation fabric **145** generates a modified membrane potential variable V^+ for the neuron by adding the leak rate Lk to the membrane potential variable V.

The modified membrane potential variable V^+ and the internal state variable h provide the current neuronal state $S_{current}$ of the neuron. To obtain state transition information for the current neuronal state $S_{current}$, the lookup table **130** is addressed by combining the most significant bits of the modified membrane potential variable V^+ with the most significant bits of the internal state variable h.

The lookup table **130** provides an updated membrane potential variable V' and an updated internal state variable h', wherein the variables V' and h' identify a next neuronal state S_{next} that the current neuronal state $S_{current}$ transitions to. The controller unit **110** writes the updated membrane potential variable V' and the updated internal state variable h' to the memory device **120**.

The lookup table **130** also provides an outgoing spike event if the current neuronal state $S_{current}$ generates a spike event.

The membrane potential variable V for the neuron may be reset to zero if the current neuronal state $S_{current}$ generates a spike event.

In another embodiment, each incoming spike event received is buffered. For each neuron of the system **100**, the controller unit **110** sends a read request to the memory device **120** for neuronal state information corresponding to said neuron. Based on the neuronal state information for said neuron, the computation fabric **145** integrates all incoming spike events targeting said neuron at once to generate a modified membrane potential variable V^+ for said neuron. The modified membrane potential variable V^+ and the internal state variable h for said neuron are then used to address the lookup table **130** to obtain state transition information for said neuron.

As shown in FIG. **7**, the computation circuit **145** comprises a first multiplexer **150**, an adder **160**, a demultiplexer **170**, and a second multiplexer **180**. The first multiplexer **150** selects between a leak rate Lk and a synaptic weight for a type of synaptic connection (syn[type]). The first multiplexer **150** selects a synaptic weight for a synaptic connection when the system **100** integrates an incoming spike event. The first multiplexer **150** selects a leak rate Lk when the system **100** updates the current neuronal state $S_{current}$ of a neuron (i.e., after all incoming spike events have been integrated).

The adder **160** applies a synaptic weight for a type of synaptic connection when the system **100** integrates an incoming spike event. The adder **160** applies a leak rate Lk to a membrane potential variable V when the system **100** updates the current neuronal state $S_{current}$ of a neuron.

The demultiplexer **170** provides a modified membrane potential variable V^+ computed by the adder **160** to the second multiplexer **180** and the lookup table **130**. To obtain state transition information for a current neuronal state $S_{current}$, the lookup table **130** is addressed by combining the most significant bits of the modified membrane potential variable V^+ and the most significant bits of the internal state variable h.

The second multiplexer **180** selects between a modified membrane potential variable V^+ provided by the demultiplexer **170** and an updated membrane potential variable V' provided by the lookup table **130**. The second multiplexer **180** selects a modified membrane potential variable V^+ when the system **100** integrates each incoming spike event received. The second multiplexer **180** selects an updated membrane potential variable V' when the system **200** updates the current neuronal state $S_{current}$ of a neuron.

In one example implementation, the memory device **120** maintains neuronal state information for about 256 neurons. The size of the memory device **120** is about 256×60 bits, wherein each neuron has 60 bits of corresponding neuronal state information. In another example implementation, the size of the memory device **120** is about 256×150 bits, wherein each neuron has 150 bits of corresponding neuronal state information.

In one example implementation, the lookup table **130** is addressed by combining the 5 most significant bits of the membrane potential variable V and the 5 most significant bits of the internal state h. The lookup table **130** maintains state transition information for about 1024 neuronal states S of a neural model 1. The size of the lookup table **130** is about 1024×11 bits, wherein each neuronal state S has 11 bits of corresponding state transition information. The corresponding state transition information for each neuronal state S includes 5 bits of an updated membrane potential variable V', 5 bits of an updated internal state variable h', and 1 bit indicating whether an outgoing spike event is generated, wherein V' and h' express a next neuronal state S_{next}.

Table 2 below provides example pseudo code, demonstrating the execution of the system **100**.

TABLE 2

```
// Step 1: Handle all synaptic inputs
for n=0 to num_synaptic_inputs - 1
    if spike(n) then nLUT[n].state.V += syn[type];
// Step 2: Sync: Update all neurons' state
for n=0 to num_neurons - 1
    nLUT[n].state.V += nLUT[n].Lk;
    nLUT[n].state = dLUT[ nLUT[n].state ];
```

FIG. **8** illustrates example neuronal state information maintained in the memory device **120** in FIG. **7**, in accordance with an embodiment of the invention. As stated above, the memory device **120** maintains neuronal state information for multiple neurons. The memory device **120** is organized into multiple entries **121**. Each entry **121** maintains neuronal state information for a corresponding neuron.

FIG. **9** illustrates example neuronal state information maintained in an entry **121** of the memory device **120** in FIG. **8**, in accordance with an embodiment of the invention. In one embodiment, each entry **121** corresponding to a neuron maintains at least one variable of a neural model 1, such as a membrane potential variable V and an internal state variable h. The membrane potential variable V of a neuron and the internal state variable h of the neuron provides the current neuronal state $S_{current}$ of the neuron.

Each entry **121** corresponding to a neuron may further include neuron parameters, such as a leak rate Lk of the neuron, synaptic weights for different types of synaptic connections (e.g., Syn0, Syn1, Syn2, Syn3), and other neuron parameters for complex neural models.

FIG. **10** illustrates example state transition information maintained in the lookup table **130** in FIG. **7**, in accordance with an embodiment of the invention. As stated above, the lookup table **130** maintains state transition information for a finite number of neuronal states S. The lookup table **130** is organized into multiple entries **131**. Each entry **131** maintains state transition information for a corresponding neuronal state S. The state transition information for each neuronal state S identifies a next neuronal state S_{next} (i.e., subsequent neuronal state) that said state s transitions to.

The lookup table **130** is addressed by a neuronal state S. As stated above, a neuronal state of a neuron may be expressed using a membrane potential variable V and an internal state variable h of the neuron. As such, in one embodiment, the lookup table **130** is addressed by combining the most significant bits of a membrane potential variable V with the most significant bits of an internal state variable h.

FIG. **11** illustrates example state transition information maintained in an entry **131** of the lookup table **130** in FIG. **10**, in accordance with an embodiment of the invention. In one embodiment, each entry **131** corresponding to a neuronal state S identifies a next neuronal state S_{next} that said neuronal state S transitions to. As stated above, in one embodiment, a neuronal state S may be expressed using a membrane potential variable V and an internal state variable h. Therefore, each entry **131** maintains a membrane potential variable V_{next} and an internal state variable h_{next}, wherein V_{next} and h_{next} express a next neuronal state S_{next}.

Each entry **131** may further maintain spiking information indicating whether a corresponding neuronal state S generates an outgoing spike event.

FIG. **12** illustrates an example neural network system **200** for simulating neurons of a neural model 1, wherein the system **200** includes a delta-based lookup table **230** that

maintains high-resolution delta values of at least one variable of the neural model 1, in accordance with an embodiment of the invention. The system **200** comprises a controller unit **110**, a memory device **120**, a delta-based lookup table **230**, a computation fabric **145**, a sync module **240**, a first adder **295**, and a second adder **296**.

The memory device **120** is a memory array that maintains neuronal state information for multiple neurons of the same neural model 1 (i.e., neurons of the same type). The neuronal state information for a neuron includes the current neuronal state $S_{current}$ of the neuron (i.e., the neuronal state of the neuron at the current time step). A neuronal state S of a neuron may be expressed using one or more variables of the neural model 1, such as a membrane potential variable V and an internal state variable h. The neuronal state information for a neuron may further include neuron parameters, such as a leak rate Lk and synaptic weights for different types of synaptic connections (e.g., Syn0, Syn1, Syn2, Syn3).

The controller unit **110** manages all synaptic inputs received. In one embodiment, the controller unit **110** receives incoming spike events targeting neurons of the system **200**. Each incoming spike event is tagged with information identifying a target neuron for said spike event, and a type of synaptic connection that said spike event was received from.

The computation fabric **145** integrates each incoming spike event received.

The sync module **240** generates synchronization signals for the controller unit **110**. The controller unit **110** sequentially updates the current neuronal state $S_{current}$ of each neuron of the system **200** when the controller unit **110** receives a synchronization signal.

The delta-based lookup table **230** maintains state transition information for a finite number of neuronal states S. Specifically, the delta-based lookup table **230** maintains high-resolution delta values of one or more variables of the neural model 1. The high-resolution delta values maintained in the delta-based lookup table **230** allows the system **200** to capture both the slowest and the fastest neural dynamics of the neural model 1. As described in detail later herein, the system **200** determines a next neuronal state S_{next} for a current neuronal state $S_{current}$ based on the high-resolution delta values maintained in the delta-based lookup table **230** for the current neuronal state $S_{current}$.

In one embodiment, the delta-based lookup table **230** maintains high-resolution delta values of the membrane potential variable V (i.e., ΔV) and high-resolution delta values of the internal state variable h (i.e., Δh). A neuronal state S is used to address (i.e., index) the delta-based lookup table **230**. For example, the delta-based lookup table **230** is addressed by combining the most significant bits of the membrane potential variable V with the most significant bits of the internal state variable h.

For each incoming spike event received in a current time step t, the controller unit **110** sends a read request to the memory device **120** for the following neuron parameters: the membrane potential variable V of a target neuron, and the synaptic weight for type of synaptic connection that said spike event was received from. The computation fabric **145** generates a modified membrane potential variable V^+ for the target neuron by adding the synaptic weight to the membrane potential variable V (i.e., integrating the spike event). The controller unit **110** writes the modified membrane potential variable V^+ to the memory device **120**.

The memory device **120** is read out sequentially when the controller unit **110** receives a synchronization signal. For each neuron of the system **200**, the controller unit **110** sends a read request to the memory device **120** for the membrane potential variable V of the neuron, the internal state variable h of the neuron, and the leak rate Lk of the neuron. The computation fabric **145** generates a modified membrane potential variable V^+ for the neuron by adding the leak rate Lk to the membrane potential variable V.

The modified membrane potential variable V^+ and the internal state variable h provide the current neuronal state $S_{current}$ of the neuron. To obtain state transition information for the current neuronal state $S_{current}$, the delta-based lookup table **230** is addressed by combining the most significant bits of the modified membrane potential variable V^+ with the most significant bits of the internal state variable h.

The state transition information for the current neuronal state $S_{current}$ includes a high-resolution delta membrane potential variable ΔV and a high-resolution delta internal state variable Δh. The system **200** determines a next neuronal state S_{next} for the current neuronal state $S_{current}$ using the delta membrane potential variable ΔV and the delta internal state variable Δh. Specifically, the first adder **295** provides an updated membrane potential variable V' by adding the delta membrane potential variable ΔV to the modified membrane potential variable V^+. The second adder **296** provides an updated internal state variable h' by adding the delta internal state variable Δh to the internal state variable h.

The updated membrane potential variable V' and the updated internal state variable h' provide the next neuronal state S_{t+1} of the neuron. The controller unit **110** writes the updated membrane potential variable V' and the updated internal state variable h' to the memory device **120**.

The lookup table **230** also provides an outgoing spike event if the current neuronal state $S_{current}$ generates a spike event. The membrane potential variable V for the neuron may be reset to zero if the current neuronal state $S_{current}$ generates a spike event.

In another embodiment, each incoming spike event received is buffered. For each neuron of the system **100**, the controller unit **110** sends a read request to the memory device **120** for neuronal state information corresponding to said neuron. Based on the neuronal state information for said neuron, the computation fabric **145** integrates all incoming spike events targeting said neuron at once to generate a modified membrane potential variable V^+ for said neuron. The modified membrane potential variable V^+ and the internal state variable h for said neuron are then used to address the lookup table **230** to obtain state transition information for said neuron.

As shown in FIG. **12**, the computation circuit **145** comprises a first multiplexer **150**, a third adder **160**, a first demultiplexer **170**, and a second multiplexer **180**. The first multiplexer **150** selects between a leak rate Lk and a synaptic weight for a type of synaptic connection (syn[type]). The first multiplexer **150** selects a synaptic weight for a type of synaptic connection when the system **200** integrates an incoming spike event. The first multiplexer **150** selects a leak rate Lk when the system **200** updates the current neuronal state $S_{current}$ of a neuron (i.e., after all incoming spike events have been integrated).

The third adder **160** applies a synaptic weight for a type of synaptic connection to a membrane potential variable V when the system **200** integrates an incoming spike event. The third adder **160** applies a leak rate Lk to a membrane potential variable V when the system **200** updates the current neuronal state $S_{current}$ of a neuron.

The first demultiplexer **170** provides a modified membrane potential variable V^+ computed by the third adder **160** to the second multiplexer **180** and a second demultiplexer **285**. The second demultiplexer **285** provides the modified membrane

potential variable V^+ to the second adder **296** and the most significant bits of the modified membrane potential variable V^+ to the delta-based lookup table **230**. A third demultiplexer **290** provides the internal state variable h to the first adder **295** and the most significant bits of the internal state variable h to the delta-based lookup table **230**. To obtain state transition information for a current neuronal state $S_{current}$, the delta-based lookup table **230** is addressed by combining the most significant bits of the modified membrane potential variable V^+ and the most significant bits of the internal state variable h.

The second multiplexer **180** selects between a modified membrane potential variable V^+ provided by the first demultiplexer **170** and an updated membrane potential variable V' provided by the second adder **296**. The second multiplexer **180** selects a modified membrane potential variable V^+ when the system **200** integrates each incoming spike event. The second multiplexer **180** selects an updated membrane potential variable V' when the system **200** updates the current neuronal state $S_{current}$ of a neuron.

In one example implementation, the memory device **120** maintains neuronal state information for about 256 neurons. The size of the memory device **120** is about 256×60 bits, wherein each neuron has 60 bits of corresponding neuronal state information.

In one example implementation, the delta-based lookup table **230** is addressed by combining the 5 most significant bits of the membrane potential variable V and the 5 most significant bits of the internal state h. The delta-based lookup table **130** maintains state transition information for about 1024 neuronal states of a neural model 1. The size of the lookup table **130** is about 1024×21 bits, wherein each neuronal state has 21 bits of corresponding state transition information. The corresponding state transition information for each neuronal state includes 10 bits of a delta membrane potential variable ΔV, 10 bits of a delta internal state variable Δh, and 1 bit indicating whether an outgoing spike event is generated.

Table 3 below provides example pseudo code, demonstrating the execution of the system **100**.

TABLE 3

```
// Step 1: Handle all synaptic inputs
for n=0 to num_synaptic_inputs − 1
    if spike(n) then nLUT[n].state.V += syn[type];
// Step 2:Sync: Update all neuron's states
for n=0 to num_neurons − 1
    state =nLUT[n].state
    nLUT[n].state.V += dLUT[ state ].ΔV + nLUT[n].Lk;
    nLUT[n].state.h += dLUT[ state ].Δh;
```

FIG. **13** illustrates example state transition information maintained in the lookup table **230** in FIG. **12**, in accordance with an embodiment of the invention. As stated above, the lookup table **230** maintains state transition information for a finite number of neuronal states S. The lookup table **230** is organized into multiple entries **231**. Each entry **231** maintains state transition information for a corresponding neuronal state S.

The lookup table **230** is addressed by a neuronal state S. As stated above, a neuronal state of a neuron may be expressed using a membrane potential variable V and an internal state variable h of the neuron. As such, in one embodiment, the lookup table **230** is addressed by combining the most significant bits of a membrane potential variable V with the most significant bits of an internal state variable h.

FIG. **14** illustrates example state transition information maintained in an entry **231** of the lookup table **230** in FIG. **13**, in accordance with an embodiment of the invention. In one embodiment, each entry **231** corresponding to a neuronal state S provides a high-resolution delta membrane potential variable ΔV and a high-resolution delta internal state variable Δh. The system **200** determines a next neuronal state S_{next} for a neuronal state S using the delta membrane potential variable ΔV and the delta internal state variable Δh corresponding to the neuronal state S.

Each entry **231** may further maintain spiking information indicating whether a corresponding neuronal state S generates an outgoing spike event.

In one embodiment, a LUT **130**, **230** is multiplexed to service all neurons that the memory device **120** maintains neuronal state information for, such that the same LUT **130**, **230** is used to describe the neural dynamics of the neurons. The neurons may be configured to exhibit different neural dynamics by configuring at least one neuron parameter (e.g., the leak rate Lk) corresponding to each neuron.

FIG. **15** illustrates a flowchart of an example process **800** for integrating synaptic firing events and updating each neuron's state, in accordance with an embodiment of the invention. In process block **801**, the system receives incoming firing events. In process block **802**, the system determines whether a new time step has begun. If a new time step has not begun, return to process block **802**. If a new time step has begun, proceed to process block **803** wherein incoming firing events for the current time step are integrated. In process block **804**, determine whether a synchronization signal has been received. If a synchronization signal has not been received, return to process block **804**. If a synchronization signal has been received, proceed to process block **805** wherein each neuron's neuronal state is sequentially updated. After process block **805**, return to process block **801**.

FIG. **16** illustrates a flowchart of an example process **899** for mapping neural dynamics of a neural model on to a lookup table, in accordance with an embodiment of the invention. In process block **900**, a neuronal model to a two-variable neuronal model is mapped. In process block **901**, a phase plane for a neural model is defined. In process block **902**, the phase plane to obtain state transition information for multiple neuronal states is coarsely sampled. In process block **903**, the state transition information on to a lookup table is mapped.

FIG. **17** is a high level block diagram showing an information processing system **300** useful for implementing one embodiment of the invention. The computer system includes one or more processors, such as processor **302**. The processor **302** is connected to a communication infrastructure **304** (e.g., a communications bus, cross-over bar, or network).

The computer system can include a display interface **306** that forwards graphics, text, and other data from the communication infrastructure **304** (or from a frame buffer not shown) for display on a display unit **308**. The computer system also includes a main memory **310**, preferably random access memory (RAM), and may also include a secondary memory **312**. The secondary memory **312** may include, for example, a hard disk drive **314** and/or a removable storage drive **316**, representing, for example, a floppy disk drive, a magnetic tape drive, or an optical disk drive. The removable storage drive **316** reads from and/or writes to a removable storage unit **318** in a manner well known to those having ordinary skill in the art. Removable storage unit **318** represents, for example, a floppy disk, a compact disc, a magnetic tape, or an optical disk, etc. which is read by and written to by removable storage drive **316**. As will be appreciated, the removable storage unit

318 includes a computer readable medium having stored therein computer software and/or data.

In alternative embodiments, the secondary memory 312 may include other similar means for allowing computer programs or other instructions to be loaded into the computer system. Such means may include, for example, a removable storage unit 320 and an interface 322. Examples of such means may include a program package and package interface (such as that found in video game devices), a removable memory chip (such as an EPROM, or PROM) and associated socket, and other removable storage units 320 and interfaces 322, which allows software and data to be transferred from the removable storage unit 320 to the computer system.

The computer system may also include a communication interface 324. Communication interface 324 allows software and data to be transferred between the computer system and external devices. Examples of communication interface 324 may include a modem, a network interface (such as an Ethernet card), a communication port, or a PCMCIA slot and card, etc. Software and data transferred via communication interface 324 are in the form of signals which may be, for example, electronic, electromagnetic, optical, or other signals capable of being received by communication interface 324. These signals are provided to communication interface 324 via a communication path (i.e., channel) 326. This communication path 326 carries signals and may be implemented using wire or cable, fiber optics, a phone line, a cellular phone link, an RF link, and/or other communication channels.

In this document, the terms "computer program medium," "computer usable medium," and "computer readable medium" are used to generally refer to media such as main memory 310 and secondary memory 312, removable storage drive 316, and a hard disk installed in hard disk drive 314.

Computer programs (also called computer control logic) are stored in main memory 310 and/or secondary memory 312. Computer programs may also be received via communication interface 324. Such computer programs, when run, enable the computer system to perform the features of the present invention as discussed herein. In particular, the computer programs, when run, enable the processor 302 to perform the features of the computer system. Accordingly, such computer programs represent controllers of the computer system.

From the above description, it can be seen that the present invention provides a system, computer program product, and method for implementing the embodiments of the invention. The present invention further provides a non-transitory computer-useable storage medium for hierarchical routing and two-way information flow with structural plasticity in neural networks. The non-transitory computer-useable storage medium has a computer-readable program, wherein the program upon being processed on a computer causes the computer to implement the steps of the present invention according to the embodiments described herein. References in the claims to an element in the singular is not intended to mean "one and only" unless explicitly so stated, but rather "one or more." All structural and functional equivalents to the elements of the above-described exemplary embodiment that are currently known or later come to be known to those of ordinary skill in the art are intended to be encompassed by the present claims. No claim element herein is to be construed under the provisions of 35 U.S.C. section 112, sixth paragraph, unless the element is expressly recited using the phrase "means for" or "step for."

The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the invention. As used herein, the singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

The corresponding structures, materials, acts, and equivalents of all means or step plus function elements in the claims below are intended to include any structure, material, or act for performing the function in combination with other claimed elements as specifically claimed. The description of the present invention has been presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the invention. The embodiment was chosen and described in order to best explain the principles of the invention and the practical application, and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated.

What is claimed is:

1. A neural network system, comprising:

a memory device that maintains neuronal states for multiple neurons;

a lookup table that maintains state transition information for multiple neuronal states, wherein said state transition information for said multiple neuronal states includes a sequence of neuronal states, and wherein said sequence identifies, for at least one neuronal state of said multiple neuronal states, a subsequent neuronal state that said at least one neuronal state transitions to; and

a controller unit that manages said memory device, wherein said controller unit updates a neuronal state for each neuron based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

2. The neural network system of claim 1, wherein:

said neurons are based on a neural model; and

for each time step, said controller unit updates a neuronal state for each neuron.

3. The neural network system of claim 2, wherein:

a neuronal state transitions to a subsequent neuronal state in response to changes in at least one variable of said neural model.

4. The neural network system of claim 3, wherein:

said memory device has multiple entries, wherein each entry maintains a neuronal state for a corresponding neuron.

5. The neural network system of claim 4, wherein:

said lookup table is addressed by a neuronal state.

6. The neural network system of claim 5, wherein:

said lookup table has multiple entries, wherein each entry corresponds to a neuronal state; and

each entry maintains state transition information for a corresponding neuronal state, wherein said state transition information identifies a subsequent neuronal state for said neuronal state.

7. The neural network system of claim 5, wherein:

said lookup table has multiple entries, wherein each entry corresponds to a neuronal state; and

each entry maintains state transition information for a corresponding neuronal state, wherein said state transition

information comprises at least one high-resolution delta value corresponding to a variable of said neural model.

8. The neural network system of claim **7**, wherein:

the system adds each delta value for a neuronal state to said neuronal state to obtain a subsequent neuronal state.

9. The neural network system of claim **5**, wherein:

said lookup table further comprises spiking information for multiple neuronal states.

10. The neural network system of claim **5**, wherein:

a neuronal state includes at least one variable of said neural model.

11. The neural network system of claim **10**, wherein:

said lookup table is addressed by combining two or more variables of said neural model.

12. The neural network system of claim **1**, wherein the system further comprises:

a computation circuit for integrating each incoming spike event received.

13. An apparatus, comprising:

a memory device including a lookup table that maintains information about multiple system states of a model system, wherein said lookup table is configured to provide information relating to a change in a system state when addressed by a current system state of said model system; and

a controller unit configured to update system states for multiple instances of said model system based on said information maintained in lookup table and incoming data received by said controller unit.

14. The apparatus of claim **13**, wherein:

for each system state in said lookup table:

said system state is represented as an n-bit digital number in said lookup table, wherein n is a natural number; and

information relating to a change in said system state is represented as an m-bit digital number in said lookup table, wherein m is a natural number;

wherein, for said current system state of said model system, said controller unit is further configured to add incoming data and an m-bit digital number representing a change in said current system state to an n-bit digital number representing said current system state to determine a next system state for said current system state.

15. The apparatus of claim **14**, wherein:

said controller unit is further configured to add incoming data and a subset of

said m-bit digital number representing a change in said current system state to a subset of said n-bit digital number representing said current system state to determine a next system state for said current system state.

16. The apparatus of claim **13**, wherein:

for each system state in said lookup table:

said system state is represented as an n-bit digital number in said lookup table, wherein n is a natural number; and

a next system state for said system state is represented as an m-bit digital number in said lookup table, wherein m is a natural number;

wherein, for said current system state of said model system, said controller unit is further configured to combine incoming data and an n-bit digital number representing said current system state to determine a next system state for said current system state.

17. The apparatus of claim **13**, wherein:

said model system is a neural system.

18. The apparatus of claim **17**, wherein:

for each system state in said lookup table, said lookup table further includes additional information about whether said model system is spiking at said system state.

19. The apparatus of claim **13**, wherein:

for each system state in said lookup table, said lookup table further includes additional metadata information about said system state.

20. The apparatus of claim **13**, further comprising:

an integrated circuit that includes both said memory device and said controller unit.

21. The apparatus of claim **13**, further comprising:

a first integrated circuit that includes said memory device; and

a second integrated circuit that includes said controller unit;

wherein said first and said second integrated circuits are interconnected.

22. A method, comprising:

maintaining neuronal states for multiple neurons;

maintaining state transition information for multiple neuronal states in a lookup table, wherein said state transition information for said multiple neuronal states includes a sequence of neuronal states, and wherein said sequence identifies, for at least one neuronal state of said multiple neuronal states, a subsequent neuronal state that said at least one neuronal state transitions to; and

updating a neuronal state for each neuron based on incoming spike events targeting said neuron and state transition information corresponding to said neuronal state.

23. The method of claim **22**, wherein:

said neurons are based on a neural model; and

a neuronal state transitions to a subsequent neuronal state in response to changes in at least one variable of said neural model.

24. The method of claim **23**, further comprising:

addressing said lookup table by a neuronal state.

25. The method of claim **24**, further comprising:

for each neuronal state, maintaining state transition information identifying a subsequent neuronal state that said neuronal state transitions to.

26. The method of claim **24**, further comprising:

for each neuronal state, maintaining state transition information comprising at least one high-resolution delta value corresponding to a variable of said neural model;

wherein each delta value for a neuronal state is added to said neuronal state to obtain a subsequent neuronal state that said neuronal state transitions to.

27. A method, comprising:

maintaining system states for multiple instances of a model system;

maintaining system state change information for said system states in a lookup table, wherein said system state change information comprises a sequence of system states, and wherein said sequence identifies, for at least one system state of said system states, a subsequent system state that said at least one system state transitions to; and

for each instance of said model system, updating said system state for said instance based on incoming data and said system state change information maintained.

* * * * *